

The Law of Unintended Consequences: The Case of External Subgoal Support

J. Gregory Trafton

Naval Research Laboratory
Washington, DC
greg.trafton@nrl.navy.mil

Raj M. Ratwani

MedStar Institute for Innovation
Washington, DC
raj.ratwani@medicalhfe.org

ABSTRACT

Many interfaces have been designed to prevent or reduce errors. These interfaces may, in fact, reduce the error rate of specific error classes, but may also have unintended consequences. In this paper, we show a series of studies where a better interface did not reduce the number of errors but instead shifted errors from one error class (omissions) to another error class (perseverations). We also show that having access to progress tracking (a progress bar) does not reduce the number of errors. We propose and demonstrate a solution – a predictive error system – that reduces errors based on the error class, not on the type of interface.

Author Keywords

Error Prediction; Progress Tracking; Interface subgoal support; Computer Human Interaction

ACM Classification Keywords

H.5.2. User Interfaces

H.1.2. User/Machine Systems

INTRODUCTION

When performing a routine procedural task, people occasionally make errors despite having the correct knowledge of how to perform the task and despite being well practiced on the task. These procedural errors, also called skill-based errors [1], generally have a low base rate, and occur less than five percent of the time [2; 3]. However, in high risk domains like aviation, medicine, nuclear energy, or military systems, procedural errors can have disastrous consequences [4; 5].

An interesting finding from researchers who have studied routine procedural tasks is that the very last step of a procedure is especially error-prone [2; 6-10]. Forgetting to include an attachment to an email or leaving the original on the glass after making copies or clicking twice on a credit

card button after ordering something from an online store are all relatively common errors that occur at the end of a procedure [10-12].

Interface designers have spent a fair amount of energy attempting to minimize the probability of errors on these types of routine tasks. The typical goal is to not only prevent errors, but to also be unobtrusive about it and to provide a good user experience as well. The most popular method for preventing procedural errors is to provide interface support for what the user has accomplished; this is goal-state information.

Interface support comes in multiple methods, but we will focus on two ways that modern interfaces support goal-state tracking for the user. Gray [13] refers to the process of goal state tracking as global placekeeping and Maxion and Reeder [14] refer to this as external subgoal support. When users have to keep track of which subgoals have been accomplished, a cognitive burden is introduced by way of an increased memory load. If the interface provides explicit information about which goals have been accomplished overall, user workload can be reduced since the goal state information no longer needs to be maintained in memory.

The need for interface cues as an indicator of what the user has accomplished so far has been well documented [15] and is a general guideline that is part of most design criteria. These indicators are particularly important for well-structured, procedural tasks. For example, most interface or web guidelines have wording similar to that given by the International Organization for Standardization's Guidance on World Wide Web, "For well-defined user tasks such as purchasing a product, the navigation structure guides users through that task and gives users a clear indication of their current position within the task." We will focus on two common methods for providing subgoal support in interfaces.

As users complete work on a procedural task, simply leaving that information on the interface provides a method for users to know which components of the task have been completed. Take, for example, the task of placing an order on Amazon.com. As the user completes the various data fields, such as entering the name, address and credit card information, that information remains on the screen even as the user progresses through the task of entering additional order information. This trail of information on the interface serves as a place keeper to cue the user as to which fields

(c) 2014 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor or affiliate of the United States government. As such, the United States Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada
Copyright 2014 ACM 978-1-4503-2473-1/14/04...\$15.00.
<http://dx.doi.org/10.1145/2556288.2557422>

Report Documentation Page			Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.				
1. REPORT DATE MAY 2014		2. REPORT TYPE		3. DATES COVERED 00-00-2014 to 00-00-2014
4. TITLE AND SUBTITLE The Law of Unintended Consequences: The Case of External Subgoal Support			5a. CONTRACT NUMBER	
			5b. GRANT NUMBER	
			5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)			5d. PROJECT NUMBER	
			5e. TASK NUMBER	
			5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory ,Washington,DC,20375			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSOR/MONITOR'S ACRONYM(S)	
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited				
13. SUPPLEMENTARY NOTES Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Toronto, Canada, pp. 1767-1776, 26 Apr ??? 1 May 2014.				
14. ABSTRACT Many interfaces have been designed to prevent or reduce errors. These interfaces may, in fact, reduce the error rate of specific error classes, but may also have unintended consequences. In this paper, we show a series of studies where a better interface did not reduce the number of errors but instead shifted errors from one error class (omissions) to another error class (perseverations). We also show that having access to progress tracking (a progress bar) does not reduce the number of errors. We propose and demonstrate a solution ??? a predictive error system ??? that reduces errors based on the error class, not on the type of interface.				
15. SUBJECT TERMS				
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 10
a REPORT unclassified	b ABSTRACT unclassified	c THIS PAGE unclassified		

have been completed and which are yet to be completed. Most procedural task interfaces display information that has been entered in the data fields to prevent the user from having to remember what data fields have been completed. We call this presentation method “information trails.”

A more explicit method of providing external subgoal support is to use a checklist or progress tracker. The progress tracker provides an explicit representation of the subgoals in the procedural task and indicates which subgoals have been completed, which are to be completed, and the order of subgoals in the task. Generally, the progress tracker is displayed at the top of the interface so that the user can clearly understand where they are in the task hierarchy.

Error Types

Our primary focus in this paper is on errors that occur on the last step. Errors in procedural tasks consist of perseverations, omissions/anticipations, and intrusions [16]. Perseveration errors are repeats of a previous action [17]. For example, putting cream in a cup of coffee multiple times is a perseveration error. Omissions are skipped steps, while anticipation errors are skipped steps that are quickly rectified. For example, an omission error would be completely forgetting to put cream in a cup of coffee, while an anticipation error would be attempting to pour from an unopened container. It can be quite difficult to differentiate omission and anticipation errors [18]. Intrusion errors (sometimes called capture errors) occur when an action comes from a different, usually related, task. For example, a capture error would occur when attempting to make coffee a person gets distracted by a tea bag and instead makes tea. In this paper, we will focus on omission errors and perseveration errors that occur at the last step of a well-learned procedure. We chose to focus on the last step of the procedure in this paper because, while errors are quite rare on well-learned tasks, they are more common on the last step [2; 6-10], and it is easier to see patterns and generate statistics.

Unintended Consequences

One possible concern about “improving” the interface is that those improvements can have unintended consequences. For example, in recent years the medical field has added computerized patient order entry forms and these systems and interfaces have saved hospitals money and patient's lives [19]. However, dosage errors [19] and patient identification errors have increased [20; 21].

Many times these unintended consequences are difficult to spot empirically because researchers focus on specific variables and sometimes do not look at how an interface may shift errors from one class to another or decrease performance in an “unrelated” area. In this paper we show how improving or changing the interface not only can have unintended consequences (in our case, shifting errors from one class to another), but also propose and demonstrate a

solution that reduces errors based on the type of error, not on the type of interface.

EXPERIMENT 1

When tasks are routine, people make very few errors on them because those tasks have been performed hundreds or thousands of times. To allow meaningful analysis of the data, different researchers have examined errors by making the task difficult to remember [22; 23], added a secondary working memory task [7; 8] or interrupted participants during the routine task [24]. We used an interruption paradigm because interruptions have high ecological validity and have been shown to increase error rates even on well-learned tasks [2; 9; 25].

Experiment 1 focused on information trails. In the “No Information Trails” condition, no external subgoal support was available to participants. As participants entered order information into the data fields of the interface, the information was removed from the data fields upon completion of the subgoal. Thus, participants in this condition were forced to remember which subgoals had been completed. In the “Information Trails” condition, external subgoal support was provided by displaying the information that was entered into the data fields of each subgoal allowing participants to use this information to track their progress in the task. We should note that no designer would use the “No Information Trails” interface and that it was used here as an experimental control.

Method

Participants

63 undergraduate students participated in the experiment for course credit.

Materials

The primary task was a complex financial management task. The goal of the task was to successfully fill clients’ orders for different types of stocks. The orders were to either buy or sell a stock and were presented four at a time at the top of the screen (see Figure 1). The current prices of the stocks associated with the orders were presented in the center of the screen in the Stock Information column. The actual stock price fluctuated every 45 s.

To complete an order, participants first had to determine which of the client orders was valid by comparing the client’s requested price with the actual market price of the stock from the Stock Information column. Once an order was determined to be valid, the participant clicked the Start Order button for the respective stock. To actually fill the order, the participant had to enter details from the order itself and the Stock Information column into eight different modules on the screen. Participants had to follow a specific procedure to complete the order; the specific sequence was as follows: Quantity, Cost, Order Info, Margin, Stock Exchanges, Transaction, Stock Info, and Review. The

spatial layout of the interface is intuitive (working down the left column and then the right column of Figure 1) and made sense to participants when asked during post-experiment interviews.

After entering information in each module, the participant clicked the Confirm button and could then move on to the next module. The Confirm button used the standard interface button practice of “blinking” after it was pushed to let the user know that their mouse click had been recorded by the interface. As part of training, participants were shown that the button blink occurred any time the button was clicked. After clicking Confirm on the final module (the Review module), a pop-up window appeared confirming the details of the order. The participant then had to acknowledge the window by clicking Ok. Finally, to complete the order the participant clicked the Complete Order button (upper right corner).¹

All of the information required to complete the task was directly available on the task interface. If a participant attempted to work on a module or clicked a button that deviated from the strict procedure, the computer emitted a beep signaling that an error was made. The participant then continued working on the task until the correct action was completed.

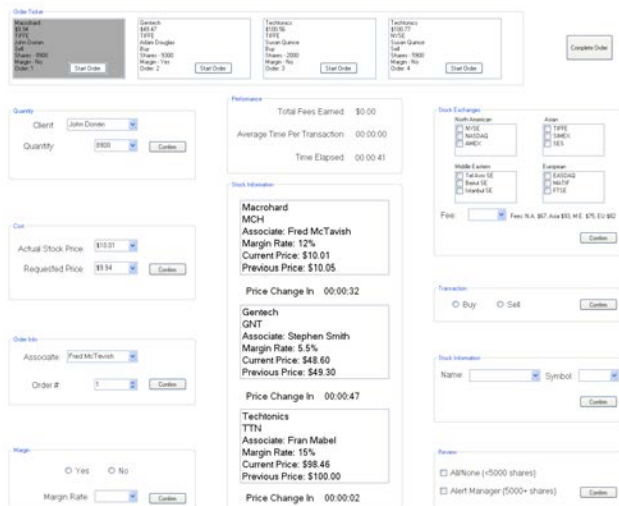


Figure 1. Screenshot of the financial management task with information trails.

The interrupting task consisted of multiple choice addition problems. Each problem contained five single-digit addends and five possible solutions (4 incorrect, 1 correct). A single addition problem and solution set was presented at one time; participants completed as many problems as possible during the interruption.

Design

Participants were randomly assigned to one of two conditions in a between-participants design. In the No Information Trails condition ($N = 33$), there were no explicit methods for external subgoal support provided. When information was entered into each module it disappeared after the confirm button was pressed. In the Information Trails condition ($N = 30$) information that was entered in each module remained after the confirm button was pressed.

In each condition, non-interruption and interruption trials were manipulated in a within-participants design. The completion of one order on the financial management task constituted a trial. Participants completed 12 trials; six were non-interruption and six were interruption trials. The order of non-interruption and interruption trials was randomized. Each interruption trial contained two interruptions. There were eight possible interruption points. These points occurred after clicking the Confirm button following the first seven modules and after acknowledging the order, just prior to the last action. The location of the interruptions was randomized with the constraint that exactly two interruptions occurred just prior to the last step and at least one interruption occurred at each of the other seven possible locations. The interruption itself lasted for 15 s.

Procedure

Participants were seated approximately 47 cm from the computer monitor. After the experimenter explained the financial management and interrupting tasks to the participant, the participant completed two training trials (one with and one without interruptions). To begin the experiment, participants had to complete two consecutive error-free trials to ensure the task was well learned.

Each participant was instructed to work at his / her own pace. When performing the interrupting task, participants were instructed to answer the addition problems as soon as the solution was known.

Error Categorization

Mouse-click data were recorded to determine the types of errors that were made on the last step (clicking the “Complete Contract” button). Error actions at the last step were categorized as either omissions or perseveration errors. An omission error was defined as skipping the step of clicking the Complete Contract button and making an action that is related to a new order on the financial management task (e.g., erroneously attempting to click the Start Order button or attempting to work on the first module). A perseveration error was defined as repeating an action on an already completed subgoal (e.g. clicking on the confirm button in the review module).

For each participant in all conditions, the percent of omission and preservation errors on the last step was

¹ Previous work has shown that the location of the complete order button does not impact error rates (Trafton, in preparation).

calculated as a ratio of the actual number of each error type to the opportunities to make an error.

Results

We only analyzed the error rate at the last step of the procedure – the complete contract button. This was done because errors at the last step are more common than mid-trial errors and it is easier to see patterns and generate informative statistics.² In addition to the traditional ANOVA results below, we also performed analyses using count data and poisson distributions; the results were quite similar, so we present the ANOVA results that are more familiar to most readers.

Interruptions

Our first analysis examined the effect of interruptions. Consistent with other research [26-31], participants made more errors on the last step when it was preceded by an interruption ($M = .31$) than when it was not preceded by an interruption ($M = .008$), $F(1, 59) = 67.8$, $MSE = 0.04$, $p < 0.05$. However, interruption did not interact with any other variable (all $p > 0.05$), so it will be collapsed for the remainder of the analyses.

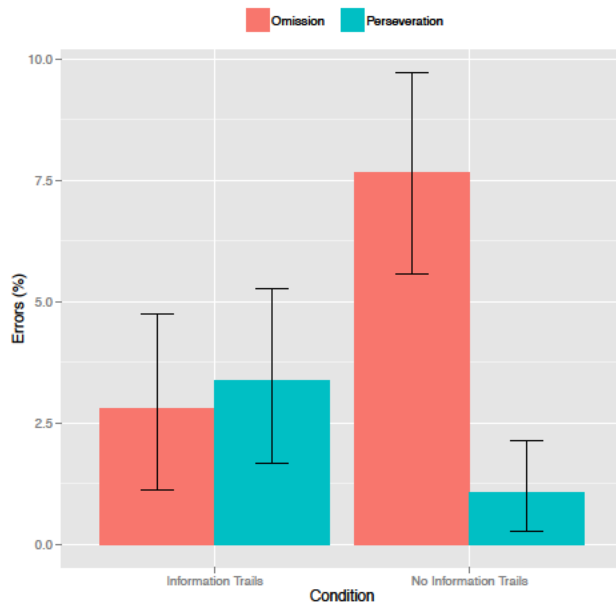


Figure 2. The error % in Experiment 1 for both omission errors and perseveration errors. Error bars are 95% CI.

Overall Error Rates

Somewhat surprisingly, there was not a difference between conditions in terms of overall error rate, $F(1, 61) = 2.5$, $MSE = 0.002$, $p > 0.10$. However, as Figure 2 suggests, there were overall more omission errors than perseveration

errors, $F(1, 61) = 10.5$, $MSE = 0.003$, $p < 0.05$. More importantly, error type interacted with condition: the error rate for both omissions and perseverations was about the same for information trails, but when there was no information trail, omissions were much higher than perseveration errors, interaction $F(1, 61) = 13.2$, $MSE = 0.003$, $p < 0.05$; see Figure 2.

Discussion

If we were to look only at one specific type of errors – omissions – we would conclude that information trails greatly reduced errors. Indeed, as Figure 2 shows, information trails reduced omission errors approximately three fold. However, a more detailed analysis shows that this is an incomplete story. An interface with information trails does not actually reduce the number of errors.³ Rather, information trails change the class of errors at the last step. Without information trails, most of the errors are omission errors. However, with information trails, both omission and perseveration errors occur with approximately equal frequency.

Why would having information trails shift the class of errors from omission errors to perseveration errors? Previous researchers have suggested that the last step of the procedure may be skipped because people think they have already completed the last step [2;7;10]; this will obviously occur when there are fewer environmental cues. When there are environmental cues (e.g., information trails), people do seem to fall into a pattern of making perseveration errors [32; 33]. Thus, information trails shift the class of errors from omission errors to perseveration errors, perhaps because information cues make it easier to remember that the last step has not yet been completed, but decay and interference still cause some perseveration errors.

Note that overall error rates were quite low: less than 10%. This is confirmatory evidence that users knew and understood the task. Also, our view is that it is almost impossible to completely eliminate errors from any given task that a person is performing; our goal is to keep overall error rates $< 1\%$. This study suggests that information trails shift errors from omission to perseveration but do not reduce the number of errors.

EXPERIMENT 2

Experiment 1 showed that information trails did not reduce the number of errors, but merely shifted the class of errors from omission errors to perseveration errors. There are, of

² Many results were similar when mid-trial errors were examined, but space precludes a detailed discussion.

³ Note that there were no differences between conditions in terms of the error rate. We interpret this result cautiously, but believe it should be taken seriously because (1) the standard deviation (3.5%) is larger than the difference between conditions (1.3%) and (2) the interaction was statistically significant, suggesting that it was not a power issue. Experiment 2 also helps address this concern by replicating the finding.

course, other ways of providing external subgoal support; a common method is to provide a progress checker, typically in the form of a progress bar at the top of the interface. Experiment 2 explored whether having a progress bar and information trails could reduce the number of errors or, as in Experiment 1, merely shift errors from one class to another. Experiment 2 also provides the opportunity to replicate the somewhat surprising findings of Experiment 1 that providing information trails did not reduce the number of errors.

There were two types of subgoal support in this experiment: information trails and progress trackers. The information trails conditions were identical to that used in experiment 1. The progress tracker condition used a progress bar that was modeled on popular retail web sites (e.g., Amazon, Toys R Us) and was displayed at the top of the interface so that the user could clearly understand where they are in the task hierarchy.

Method

Participants

64 undergraduate students participated in the experiment for course credit. No one in experiment 2 participated in experiment 1.

Materials

The primary task was the financial management task used in experiment 1; the interrupting task was the same as experiment as well.

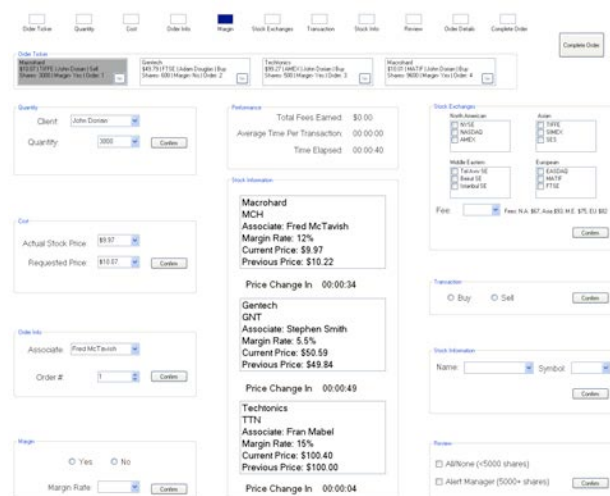


Figure 3. Screenshot of the financial management task with information trails and progress tracker.

Design

The Information Trail conditions were the same as that used in Experiment 1. The Progress Tracker conditions contained a progress bar at the top of the screen that showed the user exactly where they were in the task and what subgoal they were currently working on. The progress

tracker changed states as a step was correctly completed (the confirm button was pushed). It did not change if the user made an error. The progress tracker was always perfect information and always available to the user.

Participants were randomly assigned to one of four conditions in a between-participants design. The No Information Trails/No Progress Tracker condition ($N = 18$) contained no explicit methods for external subgoal support. The No Information Trails/Progress Tracker condition ($N = 16$) contained no information trails but did provide a progress bar. The Information Trails/No Progress Tracker condition ($N = 14$) provided information trails but no progress bar. The Information Trails/Progress Tracker condition ($N = 16$) provided full external subgoal support by providing both information trails and a progress bar.

The No Information Trail/No Progress Tracker condition was the same as the No Information Trail condition from Experiment 1. The Information Trail/No Progress Tracker condition was the same as the Information Trail condition from Experiment 1.

Procedure

The procedure for experiment 2 was the same as that for experiment 1. If a participant was in a Progress Tracker condition, the experimenter clearly explained how the progress bar worked and exactly what it showed.

To begin the experiment, participants again had to complete two consecutive error-free trials to ensure the task was well learned.

Error Categorization

Errors were categorized into omissions and perseverations in the same manner as experiment 1. Percent errors were calculated the same way as in experiment 1.

Results

Interruptions

We again examined the effect of interruptions. Consistent with experiment 1, participants made more errors on the last step when it was preceded by an interruption ($M = .18$) than when it was not preceded by an interruption ($M = .009$), $F(1, 56) = 57.9$, $MSE = 0.02$, $p < 0.05$. Interruptions did not interact with any other variable (all $p > 0.05$) and will therefore be collapsed for the remainder of the analyses.

Overall Error Rates

We analyzed the error rate for both Information Trails and the Progress Tracker. Surprisingly, but consistent with experiment 1, participants who did not have access to Information Trails did not make more errors than participants who did have access to Information Trails, $F(1, 60) < 1$, $MSE = 0.003$, NS. Also surprisingly, having access to a progress tracker did not reduce the number of errors compared to participants not having access to a progress tracker, $F(1, 60) < 1$, $MSE = 0.003$, NS.

Information Trails and Progress Tracking did not interact, $F(1, 60) < 1$, $MSE = 0.003$, NS.

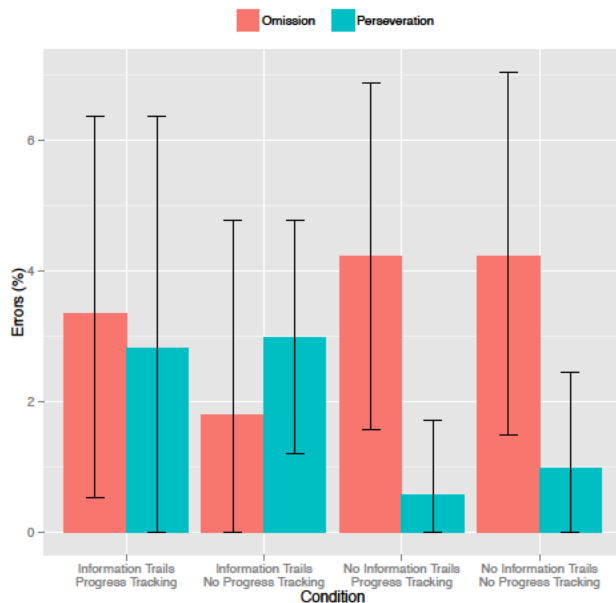


Figure 4. The error % in Experiment 2 for both omission errors and perseveration errors. Error bars are 95% CI.

As suggested by Figures 4 and 5 and consistent with experiment 1, there were more omission errors than perseveration errors, $F(1, 60) = 4.3$, $MSE = 0.002$, $p < 0.05$. Also consistent with experiment 1 and as shown in figure 5 which collapses across the Progress Tracker conditions, error type interacted with Information Trail: the error rate for omission errors were higher than perseveration errors when there was no information trail, but was about the same when there was an information trail, $F(1, 60) = 5.3$, $MSE = 0.002$, $p < 0.05$. Having access to a progress tracker did not, however, interact with the type of error, $F(1, 60) < 1$, $MSE = 0.002$, NS. Finally, there was not a three-way interaction between Information Trail, Progress Tracker, and error type, $F(1, 60) < 1$, $MSE = .002$, NS.

Discussion

This experiment examined two different types of subgoal support: information trails and progress tracking, both singly and in combination. The results were rather surprising. First, the pattern of results for information trails replicated the results of experiment 1: having information trails does not reduce the total number of errors. Experiment 2 also showed that when there was no information trail, there were more omission errors than perseveration errors on the last step of a procedure, but that when users did have access to information trails the error rate for omission errors and perseveration errors were about the same. This again shows that improving the interface may have unintended consequences: in our case it shifted

the class of errors from one type to another but did not decrease the overall rate of errors.

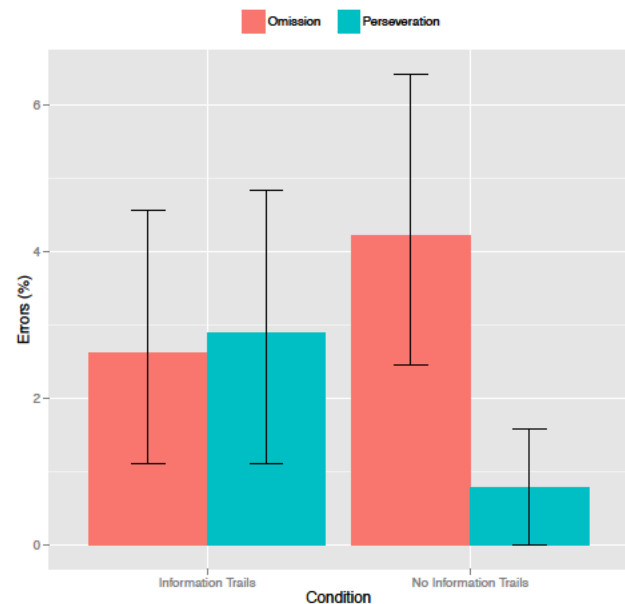


Figure 5. The error % in Experiment 2 for both omission errors and perseveration errors, collapsed across progress tracking. Error bars are 95% CI.

Another surprising result is that having a progress tracker did not reduce the number of errors at all. This is surprising because not only is a progress bar a very well known part of modern interfaces, but it showed with perfect accuracy the step that the user should have been working on. In fact, all the user had to do in order to completely eliminate errors was to look at the progress bar and execute the appropriate subgoal. The fact that the progress bar did not impact overall error rates suggests that people are not using the progress bar or perhaps not looking at it when they need it. It follows that people may not think they need it or because it required an additional gaze and was an additional level of (albeit small) work [34; 35].

One of the main aspects of experiments 1 and 2 is that people still make errors on the last step of a task, even though they know the task well. Critically, improving the interface by providing explicit subgoal support does not seem to reduce the number of errors that users make, at least on the last step of the task.

There are, however, several ways to reduce errors that could be tried. One possibility is to display only a single widget at a time and force users to fill in that information before continuing. This “single entry option” would almost assuredly reduce errors, but the user would likely be unimpressed with the interface. First, it is not uncommon to need information that has already been filled in (e.g., details of the task). Second, adding that additional step

(similar to our "Confirm" button) increases time and work for the user. Third, making changes becomes much more difficult because the user must navigate backwards.

A second way to reduce errors would be to provide explicit subgoal support by way of an explicit cue on every single step. For example, the interface could point an arrow at the step that the user was working on. Again, this approach would likely reduce errors, at least for new users. However, users are likely to become frustrated and annoyed with a cue that is always present [36]. Second, on interfaces that contain a large amount of information, a cue that is always present can contribute to visual clutter on the interface [37]. Finally, users often become accustomed to information that is always present on an interface, and users can begin to ignore this information [38]. For these reasons, a constant cue may become ineffective over a long period.

A third approach is to attempt to predict when a user is going to make an error and present a just-in-time cue only when it is likely to be needed [2]. This is the approach we will use here.

In previous work, we developed a theoretical model that predicted when a user would make an error before the error actually occurred. The model itself was based on the Memory for Goals theory [32; 39; 40] and was quite successful when converted to a real-time system. Details are available elsewhere [2], but the system was able to track a user's eye-movements and predict the likelihood that the user would make an error. When the probability reached a sufficiently high level (75% for this model), the system provided a just-in-time, blatant cue that guided the user to the next correct step. We chose this approach because it has support from different researchers [8; 41], it is not annoying because the cue is only presented when it is needed and it does not constantly increase visual clutter.

There are two important caveats to this model. First, the model was developed with interfaces that had no explicit subgoal support (neither information trails nor progress tracking), so it is unclear whether it would reduce errors on interfaces that do provide subgoal support. Second, the model was developed to work only for the last step of a procedure; it does not work for other steps in a procedure because the reason that people make errors differs according to the type of step the user is executing. However, this last caveat is actually a strength for this project: it can be used to reduce end-of-procedure errors on the current task, which is the focus of this paper.

EXPERIMENT 3

Experiments 1 and 2 showed that providing explicit subgoal support did not reduce the number of errors that users made. The goal of this paper is not only to understand how the interface impacts errors, but also to determine a method of reducing error rates on the last step of a routine procedural task.

To accomplish this, we used a real-time system that predicts when users will make an error on the last step of a procedure [2].

Method

Participants

54 undergraduate students participated in the experiment for course credit. No one in experiment 3 participated in experiment 1 or 2.

Materials

The primary task was the financial management task used in experiments 1 and 2; the interrupting task was the same as experiments 1 and 2 as well.

Design

The control condition in this experiment was the same as the Information Trails/Progress Tracker condition of experiment 2. This condition provided both Information trails and a progress bar that showed the user exactly where they were in the task and what subgoal they were currently working on. The Cue condition was exactly the same as the Information Trails/Progress Tracker, but also ran the predictive model [2].

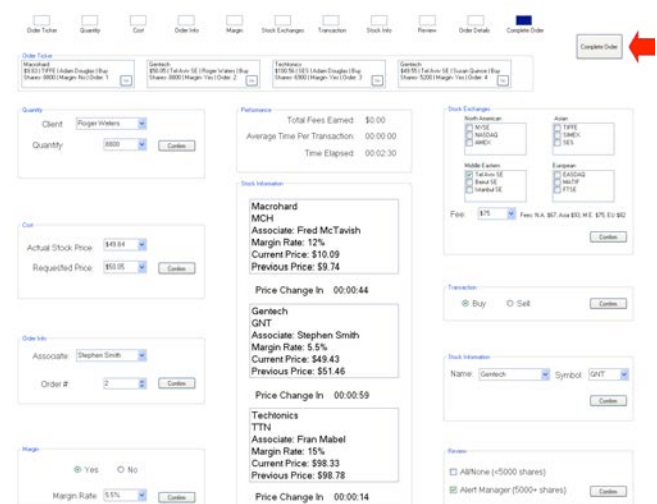


Figure 6. Screenshot of the financial management task running the predictive cue. The arrow shows that the model predicted the current user would make an error, so presented a real-time cue to the participant.

The model has three components (described more fully in [2; 10; 25]: time, the total number of fixations, and whether or not the last step was fixated. The motivation for these three variables followed directly from cognitive theory. The time predictor represents goal decay. The total number of fixations also represents decay, but it may also capture individual differences in decay rates and differences in visual and cognitive processing demands [42; 43]. Finally, the fixation on the last step represents (in)attention to the

last step and associative priming provided by the last action button on the task interface.

As the user starts working on the last step of the procedure, the predictive system calculates the probability that the user would make an error based on the aforementioned components. If the probability reached or exceeded 75% (determined through ROC analyses; see [2]), a cue would fire, alerting the user to the correct action that should be taken. The alert is shown in figure 6.

Participants were randomly assigned to one of two conditions in a between-participants design. The control condition ($N = 29$) presented full external subgoal support by providing both information trails and a progress bar. The cue condition ($N = 25$) looked and behaved identically, except that the predictive cue was shown when the probability threshold reached or exceeded 75%.

Hypotheses

Our hypothesis was that the Cue condition would reduce the overall number of errors compared to the control condition.

Procedure

The procedure for experiment 3 was the same as that for experiments 1 and 2. Participants in both conditions were calibrated on the eyetracker as well. For both conditions, the experimenter clearly explained how the progress bar worked and exactly what it showed.

To begin the experiment, participants again had to complete two consecutive error-free trials to ensure the task was well learned.

Error Categorization

Errors were categorized into omissions and perseverations in the same manner as experiment 1. Percent errors were calculated the same way as in experiment 1.

Collecting Eye Movements

Eye track data were collected using an SMI RED 250 operating at 250 Hz. A fixation was defined as a minimum of five eye samples within 30 pixels (approximately 2° of visual angle) of each other, calculated in Euclidian distance. The eyetracker was used as an input to the model to predict when a user had a high probability of making an error.

Results

Interruptions

We began by examining the effect of interruptions. Consistent with the previous two experiments, participants made more errors on the last step when it was preceded by an interruption ($M = .15$) than when it was not preceded by an interruption ($M = 0.0$), $F(1, 51) = 12.6$, $MSE = 0.03$, $p < 0.05$. As in experiments 1 and 2, interruptions will be collapsed for the remainder of the analyses.

Overall Error Rates

We first examined how well the model was able to predict when a user made an error in the control condition. Consistent with previous results, it accurately predicted 85% of the errors made.

We next analyzed the overall error rate for both the Control and Cue conditions. Consistent with our hypothesis and as suggested by Figure 7, we found that users who received the predictive cue made fewer errors than users who did not receive the predictive cue, $F(1, 52) = 12.1$, $MSE = 0.001$, $p < 0.05$.

This study also showed a larger number of perseveration errors than omission errors, $F(1, 52) = 14.43$, $MSE = 0.06$, $p < 0.05$. We also found a significant interaction between error type and condition, $F(1, 52) = 6.6$, $MSE = 0.001$, $p < 0.05$, suggesting that the Cue condition helped perseveration errors more than omission errors.

Discussion

This experiment was extremely successful: it showed that errors can be greatly reduced by running a predictive model. The predictive model successfully reduced both omission errors and perseveration errors. Critically, the model brought error rates to under 1% (see figure 7).

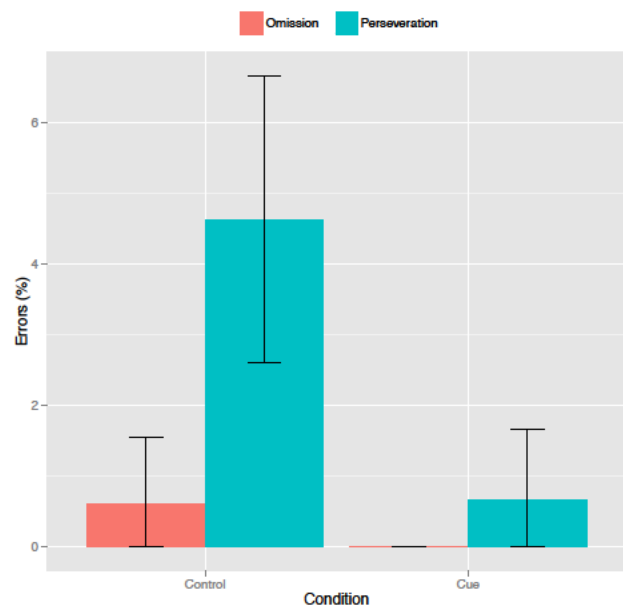


Figure 7. The error % in Experiment 3 for both omission and perseveration errors. Error bars are 95% CI.

CONCLUSION

Imagine a professional in the medical domain who has noticed that when emergency room doctors are especially busy, they sometimes forget to log off one patient and switch to a current patient, entering orders for the wrong patient [20]. This professional then requests a re-design of the interface to provide subgoal support so this omission

error (forgetting to logoff one patient) is reduced. A later study does, in fact, show that by adding subgoal support, the number of omission errors are reduced. However, an unintended consequence of this interface change was that it increased the number of times some orders got entered, so some patients had duplicate procedures performed and medications doubled (a perseveration error).

The last step of a routine procedural task frequently has one of the highest error rates [6; 9] and this paper has presented three experiments that have focused on methods for reducing the number of errors on the last step of a procedural task.

Experiment one showed that providing information trails to users did not, in fact, reduce the number of errors. Information trails did, however, have an impact on the class of errors: users who did not have information trails made more omission errors while users who did have access to information trails made approximately the same number of omission and perseveration errors. This shifting of errors from one class to another was an unintended consequence of improving the interface.

Experiment two presented two different types of external subgoal support – information trails and progress tracking. Experiment two showed that providing strong subgoal support did not decrease the number of errors. Specifically, having access to a progress bar did not reduce the number of errors, nor did having access to information trails. However, adding information trails again shifted the class of errors from omission errors to perseveration errors.

Experiments 1 and 2 suggested that, short of removing the step entirely, these end-of-procedure errors are quite resistant to interface methods of error reduction. In order to reduce the number of errors at the last step, we described a theoretical error model that predicted when people made errors on the last step of a procedure and used that model to provide a cue when the probability of making an error reached a predetermined threshold. Experiment three compared an interface with both information trails and progress tracker support to an interface with external subgoal support and the predictive cue system. Our predictive error system was able to reduce the number of errors made on the last step of the procedure to < 1%.

We believe that, as long as a person is involved, errors will occur. There are many ways to reduce or mitigate the number or type of errors that occur, including providing additional training, an excellent interface, double-checking information, etc. However all of these methods have costs in one form or another. Additional training can only reduce errors so much; double checking information (like airline pilots do) requires additional people, training, money, and errors still occur. Improving the interface can reduce the number of errors (not shown in this paper, but shown by others), but may have unintended consequences like shifting errors from one class to another.

It should be noted that in all of the current studies, the overall error rate was quite low (less than 10%). This low error rate is actually an important aspect of procedural tasks: they are well learned and thus the error rates should be low. For some tasks (e.g., making tea or making copies), making an error is either easily fixed or the consequences are not severe. However, for safety-critical tasks like nuclear power plant operations, medical procedures, airplane piloting, long-haul trucker driving, etc., errors can be disastrous. The low error rate of these types of tasks makes studying errors more challenging, but because most actions that people do are routine [3], it is critical that we understand not only why people make errors, but also how to prevent them.

The approach presented in this paper is tied to cognitive science theory and our goal is to understand why people make errors [7; 32; 39; 40; 44]. This understanding can then allow theoretically based predictive models to be built that should work across a variety of interfaces but that are focused on specific error classes (e.g., anticipation errors) or steps (e.g., the last step of a procedure). The model can then be used to predict and hopefully prevent errors as we did here. One of the advantages of class- or step- predictive models is that the interface itself may not matter as much. Thus, the interface designer can focus on creating an excellent user experience and worry less about preventing errors.

ACKNOWLEDGMENTS

This work was supported in part by the Office of Naval Research to JGT. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Navy.

REFERENCES

- [1] Rasmussen, J. 1982. Human errors. A taxonomy for describing human malfunction in industrial installations. *Journal of occupational accidents*.
- [2] Ratwani, R. M. and Trafton, J. G. 2011. A real-time eye tracking system for predicting and preventing postcompletion errors. *Human-Computer Interaction*. 26, 3, 205-245.
- [3] Reason, J. 1990. *Human Error*. Cambridge University Press.
- [4] Perrow, C. 1981. Normal accident at three mile island. *Society*. 18, 5, 17-26.
- [5] Perrow, C. 2011 *Normal accidents: Living with high risk technologies*. Princeton University Press.
- [6] Brumby, D. P., Cox, A. L., and Back, J. 2013. Recovering from an interruption: Investigating speed-accuracy trade-offs in task resumption behavior. *Journal of Experimental Psychology: Applied*
- [7] Byrne, M. D. and Bovair, S. 1997. A working memory model of a common procedural error. *Cognitive science*. 21, 1, 31-61.
- [8] Chung, P. H. and Byrne, M. D. 2008. Cue effectiveness in mitigating postcompletion errors in a routine procedural task. *International Journal of Human Computer Studies*. 66, 217-232.

- [9] Li, S. Y. W., Blandford, A., Cairns, P., and Young, R. M. 2008. The effect of interruptions on postcompletion and other procedural errors: An account based on the activation-based goal memory model. *Journal of Experimental Psychology: Applied*. 14, 4, 314-328.
- [10] Ratwani, R. M., McCurry, J. M., and Trafton, J. G. 2008. Predicting postcompletion errors using eye movements. *CHI*.
- [11] Botvinick, M. M. and Plaut, D. C. 2006. Such stuff as habits are made on: A reply to Cooper and Shallice (2006). *Psychological Review*.
- [12] Cooper, R. P. and Shallice, T. 2006. Hierarchical schemas and goals in the control of sequential behavior. *Psychological review*.
- [13] Gray, W. D. 2000. The nature and processing of errors in interactive behavior. *Cognitive Science*. 24, 2, 205-248.
- [14] Maxion, R. A. and Reeder, R. W. 2005. Improving user-interface dependability through mitigation of human error. *International Journal of Human-Computer Studies*. 63, 1, 25-50.
- [15] Hix, D. and Hartson, H. R. 1993. *Developing user interfaces: ensuring usability through product & process*. John Wiley & Sons, Inc.
- [16] Reason, J. 2000. Human error: models and management. *British Medical Journal*.
- [17] Sandson, J. and Albert, M. L. 1984. Varieties of perseveration. *Neuropsychologia*.
- [18] Cooper, R. and Shallice, T. 2000. Contention scheduling and the control of routine activities. *Cognitive Neuropsychology*.
- [19] Campbell, E. M., Sittig, D. F., Ash, J. S., Guappone, K. P., and Dykstra, R. H. 2006. Types of unintended consequences related to computerized provider order entry. *Journal of the American Medical Informatics Association*. 13, 5, 547-556.
- [20] Koppel, R., Metlay, J. P., Cohen, A., Abaluck, B., Localio, A. R., Kimmel, S. E., and Strom, B. L. 2005. Role of computerized physician order entry systems in facilitating medication errors. *JAMA: the journal of the American Medical Association*. 293, 10, 1197-1203.
- [21] Koppel, R., Wetterneck, T., Telles, J. L., and Karsh, B.-T. 2008. Workarounds to barcode medication administration systems: their occurrences, causes, and threats to patient safety. *Journal of the American Medical Informatics Association*. 15, 4, 408-423.
- [22] Giovannetti, T., Schwartz, M., and Buxbaum, L. 2007. The Coffee Challenge: A new method for the study of everyday action errors. *Journal of Clinical and Experimental Neuropsychology*. 29, 7.
- [23] Ruh, N., Cooper, R. P., and Mareschal, D. 2008. The Hierarchies and Systems that Underlie Routine Behavior: Evidence from an Experiment in Virtual Gardening. *Cognitive Science* 2008.
- [24] Botvinick, M. M. and Bylsma, L. M. 2005. Distraction and action slips in an everyday task: Evidence for a dynamic representation of task. *Psychonomic Bulletin and Review*.
- [25] Ratwani, R. M. and Gregory Trafton, J. 2010. A generalized model for predicting postcompletion errors. *Topics in Cognitive Science*. 2, 1, 154-167.
- [26] Hodgetts, H. M. and Jones, D. M. 2006. Interruption of the tower of london task: Support for a goal activation approach. *Journal of Experimental Psychology: General*. 135, 103-115.
- [27] Hodgetts, H. M. and Jones, D. M. 2006. Contextual cues aid recovery from interruption: The role of associative activation. *Journal of Experimental Psychology: Learning, Memory & Cognition*. 32, 1120-1132.
- [28] McFarlane, D. C. 2002. Comparison of four primary methods for coordinating the interruption of people in human-computer interaction. *Human Computer Interaction*. 17, 1, 63-139.
- [29] Monk, C., Boehm-Davis, D., and Trafton, J. G. 2002. The attentional costs of interrupting task performance at various stages. In *Human Factors Annual conference of 2002*,
- [30] Monk, C., Boehm-Davis, D., and Trafton, J. G. 2004. Recovering from Interruptions: Implications for Driver Distraction Research. *Human Factors*. 46, 4, 650-663.
- [31] Monk, C., Trafton, J. G., and Boehm-Davis, D. 2008. Goal decay and the resumption of suspended goals; Evidence for an activation-based theory of interruption recovery. *Journal of Experimental Psychology: Applied*. 14, 4, 299-313.
- [32] Trafton, J. G., Altmann, E. M., and Ratwani, R. M. 2011. A memory for goals model of sequence errors. *Cognitive Systems Research*. 12, 2, 134-143.
- [33] Trafton, J. G., Jacobs, A., and Harrison, A. M. 2012. Building and verifying a predictive model of interruption resumption. *Proceedings of the IEEE*. 100, 3, 648-659.
- [34] Ballard, D. H., Hayhoe, M. M., Li, F., Whitehead, S. D., Frisby, J. P., Taylor, J. G., and Fisher, R. B. 1992. Hand-eye coordination during sequential tasks [and discussion]. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*. 337, 1281, 331-339.
- [35] Gray, W. D., Sims, C. R., Fu, W. T., and Schoelles, M. J. 2006. The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review*. 113, 3, 461-482.
- [36] Ceaparu, I., Lazar, J., Bessiere, K., Robinson, J., and Shneiderman, B. 2004. Determining causes and severity of end-user frustration. *International journal of human-computer interaction*. 17, 3, 333-356.
- [37] Heiner, A. P. and Asokan, N. 2008. Using Salience Differentials to Making Visual Cues Noticeable. *UPSEC*.
- [38] Burke, M., Hornof, A., Nilsen, E., and Gorman, N. 2005. High-cost banner blindness: Ads increase perceived workload, hinder visual search, and are forgotten. *ACM Transactions on Computer-Human Interaction (TOCHI)*. 12, 4, 423-445.
- [39] Altmann, E. M. and Trafton, J. G. 2002. Memory for goals: An activation-based model. *Cognitive Science*. 26, 1, 39-83.
- [40] Altmann, E. M. and Trafton, J. G. 2007. Timecourse of Recovery from Task Interruption: Data and a Model. *Psychonomic Bulletin & Review*. 14, 6, 1079-1084.
- [41] Byrne, M. D. and Davis, E. M. 2006. Task structure and postcompletion error in the execution of a routine procedure. *Human Factors: The Journal of the Human Factors and Ergonomics Society*. 48, 4, 627-638.
- [42] Just, M. A. and Carpenter, P. A. 1976. Eye fixations and cognitive processes. *Cognitive Psychology*. 8, 4, 441-480.
- [43] Rayner, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*. 124, 372-422.
- [44] Gray, W. D. 2000. The nature and processing of errors in interactive behavior. *Cognitive Science*. 24, 2, 205-248.